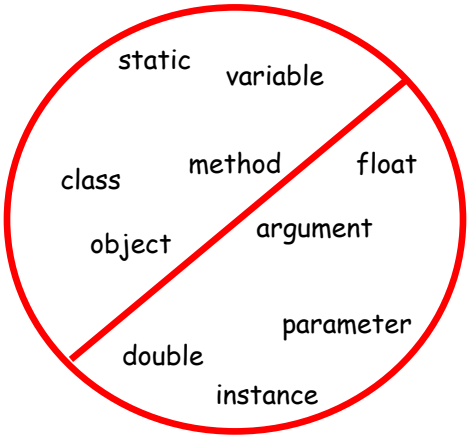
Object Oriented Programming with Buggles and Bagels

Friday, Sep. 7, 2007

CS111 Computer Programming

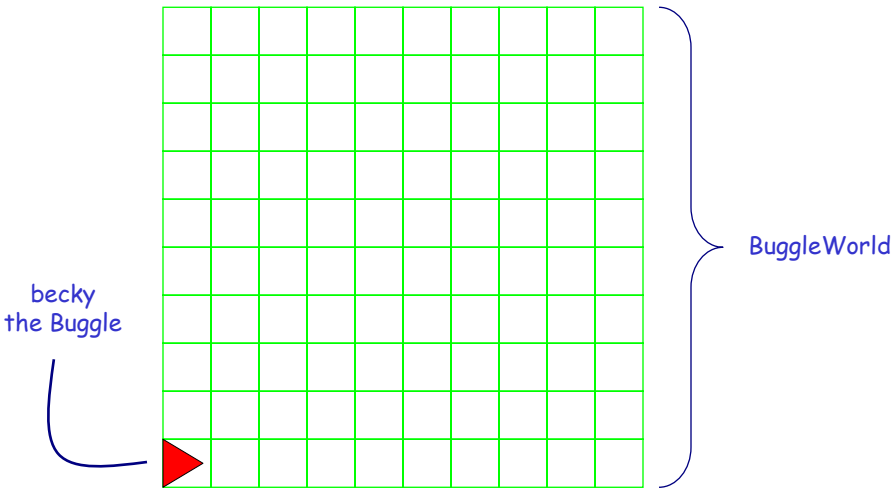
Department of Computer Science
Wellesley College

Unlearn What You Have Learned



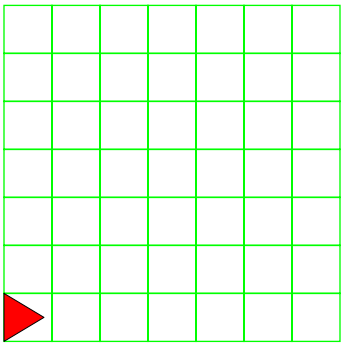
I don't think that word means what you think it means

becky in BuggleWorld



Four properties of Buggles

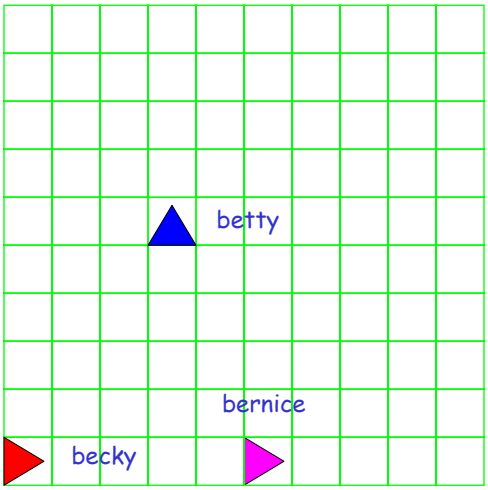
- position: Where becky sits, specified by an (x, y) coordinate.
- heading: The compass direction becky is facing.
- color: becky and her paint brush's color.
- brushDown: Is becky ready to paint?



*Collectively these four properties define the state of a Buggle.

becky has company

- A class is a collection of objects that have a common "shape" and respond the same way to a known set of messages.
- An object is an instance of a class.



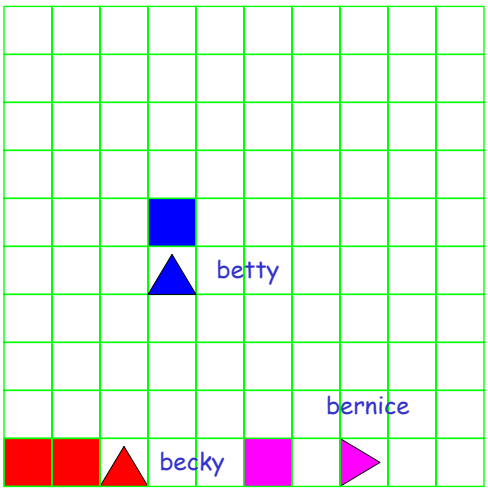
Go, becky, Go!

We change an object's state by sending it messages.

```
becky.forward();
becky.forward();
becky.left();

betty.backward();

bernice.forward();
bernice.brushUp();
bernice.forward();
```

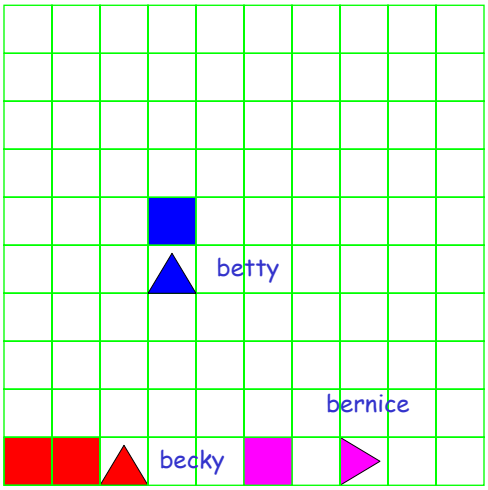


A class is described by

instance variables
that describe the properties of each class instance; and

instance methods
that are the messages to which an instance of the class can respond.

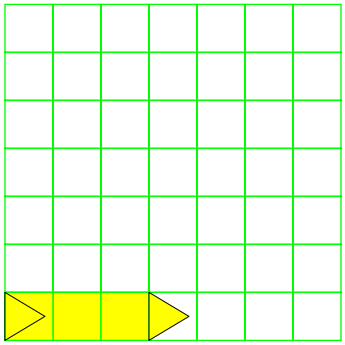
. . .



Methods with arguments (aren't we sassy?)

- Some methods require additional information when they are invoked.
- The additional information passed to the method is called an *argument*

```
becky.setColor(Color.yellow);
becky.forward(3);
```



Contracts

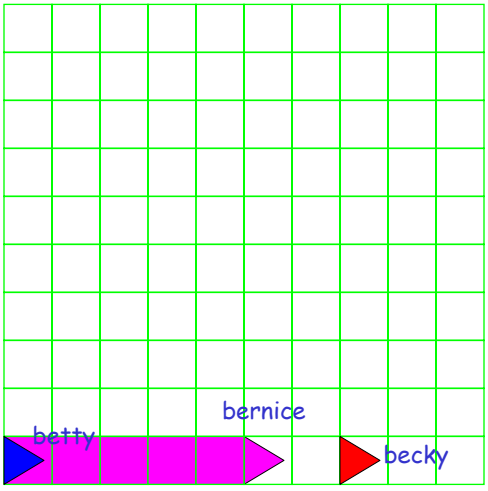
- Every class has a **contract** that specifies the behavior of its methods, i.e., how instances of the class respond to messages.
- Any user of a class can expect that objects will behave as described in the contract.
- Any implementer of the class must ensure that objects fulfill the contract.

Creating New Buggles

```
Buggle becky = new Buggle();
becky.brushUp();
becky.forward(7);

Buggle bernice = new Buggle();
bernice.setColor(Color.magenta);
bernice.forward(5);

Buggle betty = new Buggle();
betty.setColor(Color.blue);
```

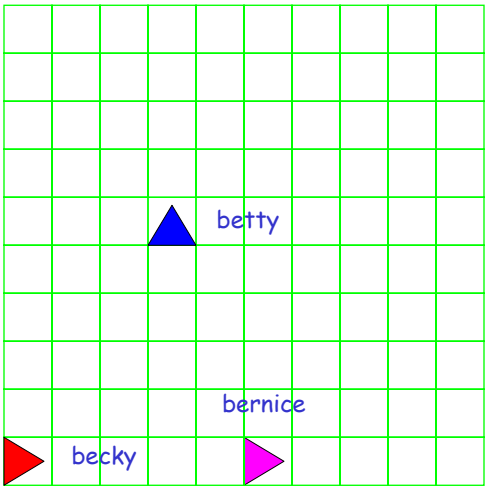


A class is described by

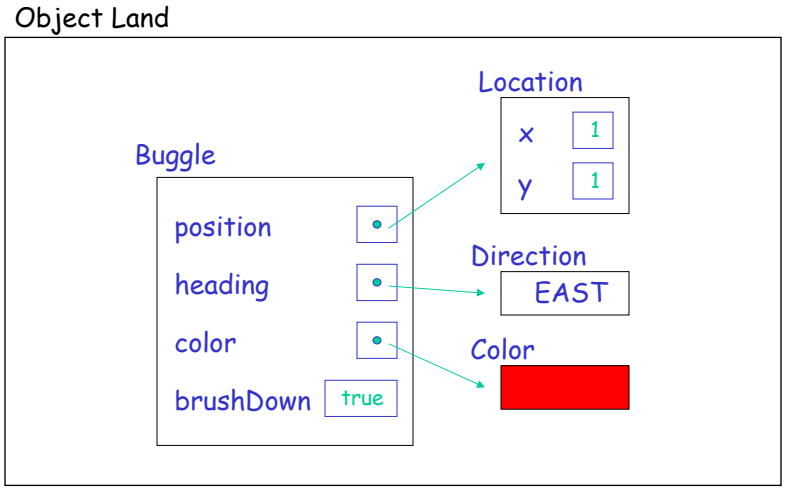
instance variables
describe the properties of each class instance;

instance methods
are the messages to which an instance of the class can respond;

constructor method(s)
create new instances of the class.



A Buggle is Born



An Example of a Class and Method

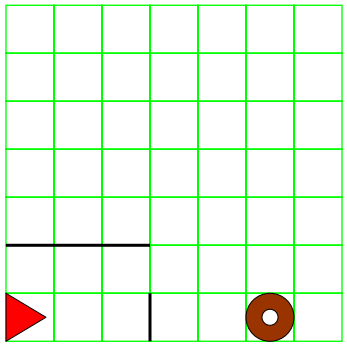
```
public class BreakfastWorld extends BuggleWorld {
  public void run () {
      Buggle becky = new Buggle();
      // becky goes outside
      becky.forward(2);
      becky.left();
      becky.forward();
      becky.right();
      becky.forward();
      becky.right();
      becky.forward();
      becky.left();
      // walks to the bagel
      becky.forward(2);
      // and chows down
      becky.pickUpBagel();
  } // run()

  ...
} // class BreakfastWorld
```

constructing a new Buggle object

run method

comments

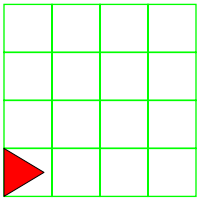


Creating becky the Buggle

③ assignment statement

```
Buggle becky = new Buggle();
```

① variable declaration ② constructor method invocation

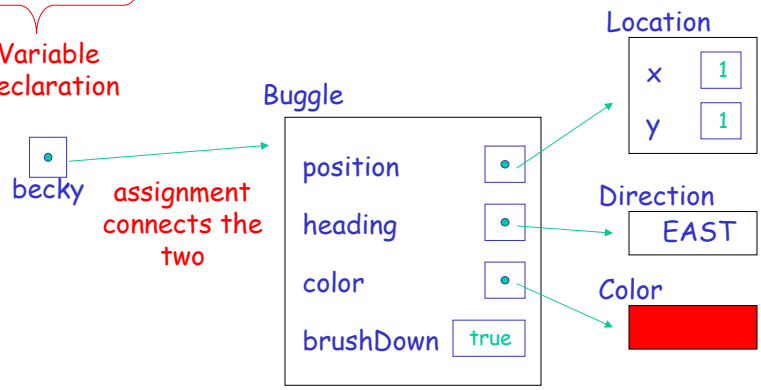


Behind the curtain

new invokes constructor method

```
Buggle becky = new Buggle();
```

Variable Declaration

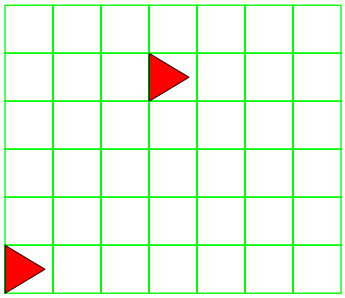


Are there classes other than Buggles?

- Yep!
- Java has many pre-defined classes
- And we will create many classes of our own throughout the semester

Witches and Wizards

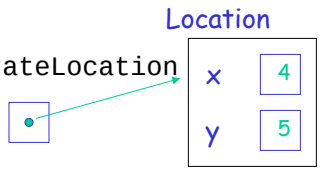
```
public class MagicWorld extends BuggleWorld {
    public void run()
    {
        Buggle harry = new Buggle();
        Location apparateLocation = new Location(4,5);
        harry.setPosition(apparateLocation);
    }
}
```



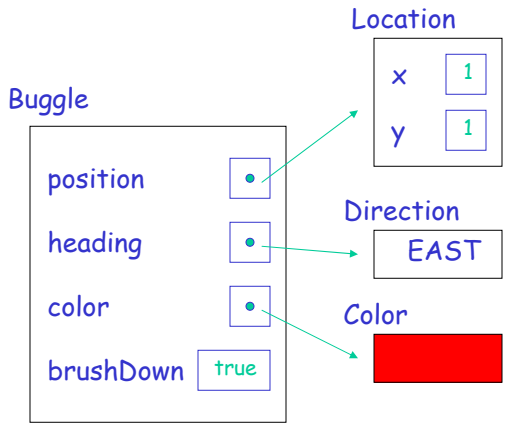
A new Location object

```
public class MagicWorld extends BuggleWorld {
    public void run()
    {
        …
        Location apparateLocation = new Location(4,5);
        …
    }
}
```

Creates a new instance of the class Location named apparateLocation

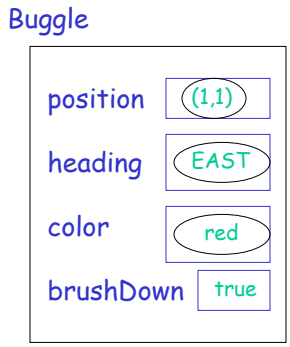


Objects inside of objects



*What about contents of variable *brushDown* inside of a Buggle object? Is this an object too?

A notational convenience



Expressions and Statements

- **Expressions** denote values and objects

```
becky
Color.yellow
3
new Buggle()
apparateLocation
new Location(4,5)
```

- **Statements** perform actions

```
becky.left();
becky.setColor(Color.yellow);
Buggle bernice = new Buggle();
bernice.forward(5);
Location apparateLocation = new Location(4,5);
harry.setPosition(apparateLocation);
```


Object-oriented Terminology

- **Object-oriented** means we create and manipulate program **objects**, which often represent things in the world (my car, you, becky the buggle).
- **Objects** are things that have **state** and can respond to **messages**. When an object receives a message, it executes the corresponding **method** — a named sequence of instructions that describes some behavior of an object.
- A **class** is a description of the shared characteristics of a group of objects. A class defines the properties (**instance variables**) that make up the state of each instance and the methods the objects understand. E.g., a buggle's color or `forward()`.
- An object created based on the class description is an **instance** of the class.
- A **constructor** method creates a new instance of a class.